REMARKS/ARGUMENTS

Claims 1, 2, 4-10, 12-18 and 20-29 are pending in the present application. Claims 7, 15, 17, and 23 were amended. No claims were added or canceled. Reconsideration of the claims is respectfully requested in view of the above amendments and the following comments.

I. 35 U.S.C. § 103, Obviousness: claims 1-2, 4-6, 9, 10, 12-14, 17, 18 and 20-22

The Examiner has rejected claims 1-2, 4-6, 9, 10, 12-14, 17, 18 and 20-22 under 35 U.S.C. § 103(a) as being unpatentable over Brook, US 2002/0038320 (hereinafter "Brook") in view of Call, US 2002/0143521 (hereinafter "Call"). This rejection is respectfully traversed.

In rejecting the claims, the Examiner states:

> Regarding claim 1, Brook teaches a method for "retrieving a data value from a character stream" by processing a text stream and obtaining information for each character in the data (text) stream (p. 9, para. 227, lines 1-5 and 231, lines 1-4). Brook teaches on performing a validity test on each character in the stream but does not clearly recite the explicit use of a data structure to store the characters by location (i.e. an array). However, in related art, Call teaches on this aspect. Call teaches the use of a data structure, an array, to store and index using integer values of character data (p. 2, para. 0016). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize a data structure like an array to index character values as demonstrated by Call in combination with the character validation method taught and suggested by Brook. One of ordinary skill in the art would have been motivated to utilize a data structure like an array to promote easy organization and efficient execution of processing functions by way of easy indexing of character values (see Call, p. 2, para. 0016). Brook teaches the use of the computer language XML (p. 9, para. 227).

Office Action dated July 18, 2007, pages 2-3.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). The prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Royka,* 490 F.2d 981, 180 USPQ 580 (CCPA 1974). In determining obviousness, the scope and content of the prior art are... determined; differences between the prior art and the claims at issue are... ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or non-obviousness of the subject matter is determined. *Graham v. John Deere Co.*, 383 U.S. 1 (1966). "Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue." *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007). "*Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.* Id. (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006))."

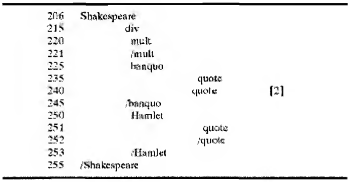
Claim 1 is as follows:

> 1. A computer-implemented character validation method comprising the steps of:
> retrieving a data value from a character stream; and
> determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language.

The combination of Brook in view of Call fails to render claims 1 obvious, as neither Brooks, nor Call, nor the combination of Brook in view of Call teaches or suggests all the features of claim 1. Specifically neither Brooks, nor Call, nor the combination of Brook in view of Call teaches or suggests the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language."

The Office Action cites to paragraph [0227] and [0231] of Brook, which are reproduced below for the Examiner's convenience, as allegedly teaching this feature.

> **[0227]** In order to better appreciate operation of the parsing process **344** as described in relation to **FIGS. 3(a)**, **3(b)** and **3(c)**, parsing of the exemplary XML fragment [1] is considered firstly in relation to the parsing process **236** described in relation to **FIG. 2.** In this case, the XML fragment [1] yields the following hierarchical representation of parsed mark-up tags in the sub-process **212**.



206	Shakespeare			
215		div		
220		mult		
221		/mult		
225		banquo		
235			quote	
240			quote	[2]
245		/banquo		
250		Hamlet		
251			quote	
252			/quote	
253		/Hamlet		
255	/Shakespeare			

[0231] Turning to **FIG. 3(b)**, the process **344** continues from "a" on the dashed boundary line **356** to a testing step **350** which determines whether a well-formedness check is to be performed. If such a check is to be performed, then the process **344** is directed in accordance with a "yes" arrow to "c" on the boundary line **358**. The dashed boundary line **358**, along with reference letters "c" to "f" is mirrored by a corresponding boundary line in **FIG. 3(c)**, in relation to which the process **344** is further described. If the well-formedness check is not to be performed, then the process **344** is directed in accordance with a "no" arrow to a testing step **352** which determines whether a validation check is to be performed. If the validation check is to be performed, then the process **344** is directed in accordance with a "yes" arrow to "e" on the dashed boundary line **358**. If, on the other hand, the validation check is not to be performed, then the process **344** is directed to "d" on the dashed boundary line **358**.

Paragraph [0227] of Brook merely teaches an example of the parser parsing an XML fragment. Paragraph [0231] of Brook explains that either a well-formedness check or a validation check is to be performed. In paragraphs [0213] and [0216], which are reproduced below for the Examiner's convenience, Brook explains that both the well-formedness test and the validation test merely verify "correct syntactic placement" of the elements:

[0213] After the step **212**, the process **236** is directed to a testing step **242**, which determines whether a well-formedness check is to be performed. Well-formedness checks ensure that the document meets appropriate "well-formedness constraints", as defined on page 5 of "Extensible Markup Language (XML) 1.0 (Second Edition) W3C Recommendation, Oct. 6, 2000", which is available on the Internet at http:.backslash..backslash.www.w3.o-rg.backslash.tr.backslash.2000.backslash.rec-xml-20001006.html. Well-formedness checks test the document for compliance with general structure rules, particularly whether tags in a document have been properly nested. If such a check is to be performed, then the process **236** is directed in accordance with a "yes" arrow to "a" on a dashed boundary line **246**. The dashed boundary line **246**, along with reference letters "a" to "d" is mirrored by a corresponding boundary line in **FIG. 2(b)**, in relation to which the process **236** is further described. If the well-formedness check is not to be performed, then the process **236** is directed in accordance with a "no" arrow from the testing step **242** to a testing step **244** which determines whether a "validation check" is to be performed. Validation checks involve a comparison of syntactic elements in a document against validity constraints defined in a Validation Reference Document (referred to as a VRD for the sake of brevity) such as a document type definition (DTD), as described in Section 5.1 of the aforementioned W3C Recommendation. DTDs and XML Schemas are examples of VRDs against which validation checks can be performed, however validation checks as described herein can be performed against other types of VRDs. This comparison procedure verifies correct syntactic placement of elements to a greater extent than the mere well-formedness check. If the validation check is to be performed, then the process **236** is directed in accordance with a "yes" arrow to "b" on the dashed boundary line **246**. If, on the other hand, the validation check is not to be performed, then the process **236** is directed in accordance with a "no" arrow to "c" on the dashed boundary line **246**.

> **[0216]** As noted, the validation checking step **220** involves a comparison of the identified syntactic element in the markup document being considered against a document type definition (DTD). This comparison procedure verifies correct syntactic placement of elements to a greater extent than the mere well-formedness check described in relation to the sub-process **238**.

Further, the validation test simply performs this "correct syntactic placement" test to a greater extent than the normal well-formedness test. (see also Figure 2b, steps 214 and 220 and Figure 3c steps 320 and 326) Paragraph [0236] of Brook describes an example of what the extended testing is that the validation test does beyond the normal well-formedness test. As explained in the example, the validation check not only checks to see if the tag pairs are properly, or fully, nested, but also checks to see if the nesting is legal. Thus, Brook teaches determining a proper syntactic structural placement of a tag or element within a structured document, such as an XML document. In contradistinction, claim 1 recites "wherein the determining step <u>determines the data value's validity as a character within a given computer language</u>." Determining a proper syntactic location of a tag or element in a structured document is not the same as <u>determining the data's validity as a character within a given computer language</u>. Therefore, Brook fails to teach the feature of "wherein the determining step determines the data value's validity as a character within a given computer language."

Further, Brook is silent as to testing a data value's validity of a character with a given computer language, Brook cannot, therefore teach or suggest the feature of "wherein the determining step determines the data value's validity as a character within a given computer language." Thus, Brooks fails to teach or suggest the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language."

Additionally, Brook does not teach the feature of "wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure." Brook fails to teach a plurality of status values. Paragraph [0237] of Brook explains that the validity test taught by Brook is performed by hashing tags and comparing the new hashes to previously generated hashes. Thus, only numbers are compared. The validation check determines if the hashes match and if the hashes are syntactically, properly located within the document structure. Thus, Brook fails teach a plurality of status values. Further, as Brook is silent in regards to a plurality of status values or using status values in a validity test, logically Brook cannot suggest a plurality of status values.

Therefore, for at least the reasons set forth above, Applicants submit that Brook fails to teach or suggest the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language."

Furthermore, Call does not cure the deficiencies of Brook. Call fails to teach or suggest the feature missing from Brook, the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language."

The Office Action admits that Brook does not teach the use of a data structure to store the characters by location. (see Office Action mailed July 18, 2007, pages 2-3.) However, the Office Action alleges that Call teaches this feature in paragraph [0016], which is reproduced below for the Examiner's convenience:

> **[0016]** It is a further object of the invention to store both fixed and variable length data as an addressable array of integer values organized to permit more efficient execution of processing functions of the type typically performed by database management systems.

All that the above passage of Call teaches is that fixed and variable length data is stored in an array. In paragraph [0018] Call explains that <u>character data which represent natural language text</u> is parsed into logical subdivisions that encapsulate the meaning of the original natural language text. These subdivisions are then replaced with a fixed length numerical integer value. As explained in paragraph [0019], the data size of each integer is smaller than the corresponding text that the integer represents. Further paragraph [0019] explains that the average English word is seven characters long plus a space, thus the integer takes up less space. Thus, Call does not teach replacing an individual character with an integer value. Rather, <u>Call teaches replacing entire natural language words with an integer</u>. In contradistinction, claim 1 recites "a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value." Thus, claim 1 recites that the member of the data structure is directly related to a data value that represents a single character. Call teaches representing entire natural language words as a single integer and storing those integers in an array. Thus, the integer stored by Call is not for a single character, nor is the data directly related to a single character, as recited in claim 1. Thus, Call fails to teach or suggest the feature of "a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value."

Furthermore, claim 1 recites that "a plurality of status values in said member of said data structure." Thus, claim 1 contemplates that the member of the data structure being examined is not only directly related to a data value that represents a single character, but also the member includes a plurality of status values. The Office Action does not specifically cite to any portion of Call as teaching this feature. Rather, the Office Action merely alleges that Call teaches use of a data structure, and array, to store and index using integer values of character data. However, Call is silent regarding any type of validity test or a member of a data structure with "a plurality of status values." Thus, Call fails to teach or suggest "a plurality of status values in said member of said data structure." In addition to the explanation set forth above regarding Brooks and "a plurality of status values," as the Office Action admits that Brook does not teach a data structure, Brook must fail to teach or suggest "a plurality of status values in said member of said data structure."

Additionally, Call does not teach the features of "wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure" and "wherein the determining step determines the data value's validity as a character within a given computer language." Call teaches changing natural language words into integers for faster, more efficient processing and storage. Call is silent in regards to determining a validity and a plurality of status values. Thus, Call must fail to teach or suggest the feature of "wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure." Further, Call actually teaches away from the feature of "wherein the determining step determines the data value's validity as a character within a given computer language," as Call teaches generating an integer for entire natural language words and not for individual characters of specific computer language.

Thus, for at least the reasons set forth above, Applicants submit that Call fails to teach or suggest the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language."

Therefore, neither Brook, nor Call, nor the combination of Brook in view of Call teaches or suggests the feature of "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein the determining step determines the data value's validity as a character within a given computer language." Thus, the Examiner has failed to properly establish a *prima facie* showing of obviousness with respect to Claim 1.

As claim 1 is representative of claims 9 and 17, the same distinctions between claim 1 and the combination of Brook in view of Call apply to these claims as well. Therefore, Applicants submit that independent claims 1, 9, and 17 are in condition for allowance over the combination of Brook in view of Call. Claims 2, 4-6, 10, 12-14, 18 and 20-22 depend from and further restrict one of independent claims 1, 9 and 17 and also patentably distinguish over the cited art, at least by virtue of their dependency.

Therefore, the rejection of claims 1-2, 4-6, 9, 10, 12-14, 17, 18, 20-22 under 35 U.S.C. § 103(a) has been overcome.

II. <u>35 U.S.C. § 103, Obviousness: claims 7-8, 15-16 and 23-25</u>

The Examiner has rejected claims 7-8, 15-16 and 23-25 under 35 U.S.C. § 103(a) as being unpatentable over Brook and Call in view of Zhao et al., US 2002/0042707 Al (hereinafter "Zhao"). This rejection is respectfully traversed.

In rejecting the claims, the Examiner states:

> Regarding claims 7 and 8, Brook teaches the use of a wide range of fonts and styles but does not explicitly disclose the use of extensible markup language (XML) syntax. However Zhao teaches the analysis and format determination of extensible markup language (XML) (see fig. 6, grammar packaging). At the time of the applicant's invention, it would have been obvious to one of ordinary skill in the art to modify Brook's method to allow it to process XML documents as input, as taught by Zhao. It logically follows that the rules employed by Brook's character validation would be in accordance with extensible markup language (XML) also. The motivation for doing so would have been to be able to determine whether extensible markup language (XML) packets match the extensible markup language (XML) protocol definition at an increased speed over prior methods. Therefore it would have been obvious to combine Brook, Call and Zhao for the benefit of increased processing speed to obtain the invention as specified in claims 7-8.
>
> Claims 15 and 16 contain similar subject matter and are rejected under the same rationale as claims 7 and 8.
>
> Claims 23 and 24 contain similar subject matter and are rejected under the same rationale as claims 7 and 8.
>
> Regarding claim 25, Brooks teaches a character validation method comprising the steps of: retrieving a data value from a character stream (p. 9, para. 227, lines 1-5 and 231, lines 1-4). Brook teaches on performing a validity test on each character in the stream but does not clearly recite the explicit use of a data structure to store the characters by location (i.e. an array). However, in related art, Call teaches on this aspect. Call teaches the use of a data structure, an array, to store and index using integer values of character data (p. 2, para. 0016). One of ordinary skill in the art at the time of the applicant's invention would have found it obvious to utilize a data structure like an array to index character values as demonstrated by Call in combination with the character validation method taught and suggested by Brook. One of ordinary skill in the art would have been motivated to utilize a data structure like an array to promote easy organization

and efficient execution of processing functions by way of easy indexing of character values (see Call, p. 2, para. 0016). Brook teaches the use of a wide range of fonts and styles but does not explicitly disclose the use of extensible markup language (XML) syntax. However Zhao teaches the analysis and format determination of extensible markup language (XML) (see fig. 6, grammar packaging). At the time of the applicant's invention, it would have been obvious to one of ordinary skill in the art to modify Brook's method to allow it to process XML documents as input, as taught by Zhao. It logically follows that the rules employed by Brook's character validation would be in accordance with extensible markup language (XML) also. The motivation for doing so would have been to be able to determine whether extensible markup language (XML) packets match the extensible markup language (XML) protocol definition at an increased speed over prior methods. Therefore it would have been obvious to combine Brook, Call and Zhao for the benefit of increased processing speed to obtain the invention as specified. Brook, Call and Zhao teach on the aspect of further comprising if each character in said character stream is valid, applying a predetermined set of syntactic rules to byte patterns comprising said character stream (Brook, para. 237, II. 6-15).

Office Action dated July 18, 2007, pages 4-6.

Claims 7-8, 15-16 and 23-24 depend from and further restrict one of independent claims 1, 9 and 17. Zhao is cited as disclosing the analysis and format determination of extensible markup language, but does not supply the deficiencies in Brook and Call as described above with respect to claim 1. Claims 7-8, 15-16 and 23-24, accordingly, patentably distinguish over the cited art, at least by virtue of their dependency.

Claim 25 as amended recites "determining a validity of a character represented by said data value by locating a member of a data structure, said member having a direct correspondence to said data value, wherein said validity is determined according to a logical combination of a plurality of status values in said member of said data structure, wherein said character stream comprises characters in accordance with a specification for an extensible markup language, and wherein a first status value of said plurality of status values indicates whether said data value represents a valid character having a first attribute corresponding to said first status value and a second status value of said plurality of status values indicates whether said data value represents a valid character having a second attribute corresponding to said second status value, wherein the determining step determines the data value's validity as a character within a given computer language." Claim 25 patentably distinguishes over the cited art for similar reasons as discussed above with respect to claim 1, and, in addition, because the cited art does not disclose or suggest "wherein a first status value of said plurality of status values indicates whether said data value represents a valid character having a first attribute corresponding to said first status value and a second status value of said plurality of status values indicates whether said data value represents a valid character having a second attribute corresponding to said second status value." Applicants respectfully disagree that any of the cited references to Brook, Call or Zhao disclose this subject matter or that such subject matter would be obvious in view of the references.

Therefore, the rejection of claims 7-8, 15-16 and 23-25 under 35 U.S.C. § 103(a) has been overcome.

III. <u>35 U.S.C. § 103, Obviousness: claims 26-29</u>

The Examiner has rejected claims 26-29 under 35 U.S.C. § 103(a) as being unpatentable over Brook, Call and Zhao, in view of Jurion et al., US 6,631,501 B1 (hereinafter "Jurion"). This rejection is respectfully traversed.

In rejecting the claims, the Examiner states:

> Regarding claims 26-29, the combination of Brook, Call and Zhao as outlined in the above rejections teaches upon the aspects of character stream parsing and performing validity tests upon the parsed characters but does not clearly teach upon the aspect wherein the parsed characters are tested to be "base" characters, "digit" characters and "extender" characters. While Brook, Call and Zhao do teach upon the usage of characters in general, nothing is explicitly recited to classify these characters into general groups (i.e. base, digit and extender). However, in related art, Jurion teaches the automatic and replacement of characters wherein characters are tested on their validity within a group or string of characters to determine whether a character within the string is appropriate, or valid. Jurion teaches that the characters analyzed can be of a plurality of different types of characters which would implicitly include "base" characters, "digit" characters, and "extender" characters as claimed by applicant and therefore one of ordinary skill in the art at the time of the applicant's invention would have found it obvious to test the validity of characters utilizing aspects taught by Jurion, specifically the use of base, digit, and extender characters (col. 3, lines 8-18). One of ordinary skill in the art would have been motivated to utilize the teachings of Jurion in combination with the teachings of Brook, Call, and Zhao in order to check the syntactical rules of character streams correctly and efficiently as provided by Jurion as a necessary need in the art of simple character validation (see Jurion, col. 2, 11, 41-52).

Office Action dated July 18, 2007, page 7.

Claims 26-29 depend from and further restrict one of the independent claims. Jurion does not supply the deficiencies in the principal references a described above. Accordingly, claims 26-29 patentably distinguish over the cited art in their present form, at least by virtue of their dependency.

In addition, Applicants continue to submit that the references do not disclose or suggest "said first status value indicates whether said data value is a valid base character, said second status value indicates whether said data value is a valid digit character, and a third status value indicates whether said data value is a valid extender character" as recited in claim 26 or similar language recited in claims 27-29; and that the claims patentably distinguish over the cited art in their own right as well as by virtue of their dependency. In particular, Jurion does not teach or suggest three different status value indicators – a first status value, a second status value and a third status value, nor the details associated with each of these three explicitly enumerated status values (the first status value indicates whether said data value is a valid base character, the second status value indicates whether said data value is a valid digit character, and a third status value indicates whether said data value is a valid extender character).

Therefore, the rejection of claims 26-29 under 35 U.S.C. § 103(a) has been overcome.

IV. Conclusion

For at least all the above reasons, this application is believed to be in condition for allowance, and it is respectfully requested that the Examiner so find and issue a Notice of Allowance in due course.

The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: December 18, 2007

Respectfully submitted,

/Gerald H. Glanzman/
Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants